

Thoroughbred
DOSLINK

Thoroughbred DOSLINK

**COPYRIGHT © 1988 CONCEPT OMEGA CORPORATION
All Rights Reserved.**



This documentation may not be reproduced or modified in any way, without the express written permission of:

Thoroughbred® Concept Omega® Corporation

19 Schoolhouse Road • PO Box 6712 • Somerset, NJ 08875-6712
Phone: 201-560-1377 • Telex: 910-3808-394 • Fax: 201-722-7958

#TRMA-051688-DLNK

Thoroughbred, Concept Omega, Thoroughbred Business BASIC and Thoroughbred Business BASIC Operating System are trademarks of Concept Omega Corporation.

Thoroughbred DOSLINK

Introduction

DOSLINK is used for accessing files and programs in the DOS partition and includes two different ways for linking to DOS:

1. ***DLNK Utility**
2. **DOSLINK Public program**

The *DLNK Utility is a user-friendly interface that allows you to copy programs or files between the TOS and DOS partitions. The DOSLINK Public Program is more technical and is provided as a tool for system programming.

1. *DLNK Utility

This utility copies individual programs or files back and forth across the Thoroughbred partition on the hard disk and the primary DOS partition on the hard disk (or a DOS floppy disk). *DLNK only supports transfers to and from the primary bootable DOS partition on the primary DOS hard disk (usually designated as drive C).

NOTE: You **MUST** have a DOS partition on your hard disk to use the *DLNK Utility. This utility does not check the integrity or compatibility of the file being transferred.

To run this utility from console mode, enter:

RUN "*DLNK" <Return>

The system displays the *DLNK - TRANSFER FILES TO/FROM DOS screen and the following prompt:

FROM WHICH PARTITION (DOS/TOS):

Enter the name of the partition from which the file will be copied. If you specify TOS, the file will be copied from the TOS partition to the DOS partition. If you specify DOS, the file will be copied from the DOS partition to the TOS partition. The system responds with the following prompt:

DISK ID (D0-D9) TO DISABLE:

Entering the Disk ID selects the physical device to be used. (The Disk ID is associated with the physical device in the *JPSD Utility or configuration of the Disk Directories in the *NPSD Utility.)

For example, if Disk ID 'D1' is chosen, and that logical disk is associated with hard disk 'H0', then hard disk 'H0' will be the physical device accessed during the copy. If Disk ID 'D9' is chosen, and that logical disk is associated with floppy disk 'F0', then floppy disk 'F0' will be the physical device accessed during the copy.

You **MUST** enter an unused Disk ID, and it **MUST NOT** be the Disk ID where the DOSLINK program resides (doing this will cause an Error 12 when the DOSLINK program is called). All other users will be denied access to that directory while this utility is being used. The system responds with the prompt:

DOS DIRECTORY PATH ('CR' = \):

Enter a standard DOS path to designate the desired DOS directory or enter <Return> to designate the ROOT (\) directory. For example, a path of \USR\PROG would access the PROG subdirectory. The system responds with the following prompt:

DOS FILENAME & EXTENSION:

Enter a standard DOS filename as described in your DOS manual. The system responds with the following prompt:

TOS FILENAME:

Enter a standard Thoroughbred BASIC filename of eight characters or less. The system responds with the following prompt:

TOS DISK ID (D0-D9):

Enter the Disk ID (D0 through D9). This refers to the TOS directory number of the 'source' file in the case of a TOS to DOS copy or the directory number of the 'destination' file in the case of a DOS to TOS copy. This directory must be currently enabled. The system responds with the following prompt:

SEND AS THOROUGHbred BASIC FILE? (Y/N):

This prompt appears for TOS to DOS copies only. Because of a difference in file structure between Thoroughbred/OS BASIC and Thoroughbred DOS BASIC, files copied from one operating system to the other will not be inherently readable or executable unless they are translated (or reformatted) during the copy. *DLNK will not translate Direct, Sort, or Serial files during a TOS to DOS copy.

This prompt does not appear during DOS to TOS copies because the utility will automatically determine whether or not the file on the DOS side is a Thoroughbred DOS BASIC file, and, if necessary, will translate it automatically. For TOS to DOS copies, a 'Y' response to this question will translate the file, an 'N' response will copy the file in its "raw" form (not translated). *DLNK will not transfer Serial files from DOS to TOS as Serial files, but as 512-byte INDEXED files.

When the transfer of the file is complete, the following prompt appears:

***FILE SUCCESSFULLY TRANSFERRED* 'CR' TO CONTINUE**

Press <Return> to transfer another file or <F4> to end the utility.

Error Messages

CAN'T FIND DOS PARTITION, PROGRAM TERMINATED

The utility returns this message if there is no DOS partition or no valid DOS boot sector on the hard disk, or if the diskette has not been formatted with the DOS FORMAT command.

CONTROL STRING CORRUPTED, PROGRAM TERMINATED

A string variable used by the utility to store essential information has become corrupted. The utility cannot continue without this information.

DISK Dx IS IN USE BY ANOTHER TERMINAL

The DISC ID chosen cannot be disabled if another terminal has one or more of that directory's files open.

DISK FULL - FILE IS NOT TRANSFERRED

The utility returns this message during a TOS to DOS transfer when there is no more available space in the DOS partition or on the floppy diskette. The portion of the file that was transferred will be erased and the space it used will be returned to the File Allocation Table.

DOS DIRECTORY IS FULL, CAN'T EXPAND IT

The utility cannot find an available entry in the specified DOS directory. This utility does not have the capability to expand a DOS subdirectory.

ERROR xx HAS OCCURRED IN DOSLINK - MODULE nnnnn

An unexpected error has occurred in the DOSLINK public program in the module specified.

ERROR xx HAS OCCURRED ON LINE xxxx

An unexpected error has occurred in the utility.

FATAL DISK ERROR, PROGRAM TERMINATED

An attempt to read or write a physical sector on the hard disk or floppy has failed. This may indicate a bad spot on the disk. This error will also be returned if an attempt is made to access a floppy drive where no floppy is present or where the floppy is not formatted.

FILE ALREADY EXISTS

The 'destination' filename for the copy already exists. The utility will not write over an existing file.

FILE NOT FOUND

The 'source' file for the copy cannot be found.

INVALID SUBDIRECTORY

A nonexistent DOS subdirectory name has been entered.

THIS PROGRAM MUST BE RUN FROM CONTROL TERMINAL ONLY

The utility returns this message if an attempt is made to run the utility from a terminal other than 'T0'.

2. DOSLINK Program

DOSLINK is a public program run from Thoroughbred/OS which reads from and writes to files in the DOS partition by physically getting and putting 512-byte sectors. DOS recognizes the concepts of sectors and files, but does not recognize the concepts of fields and records. All DOS to/from TOS data transfers are performed as 512-byte sectors or parts of sectors. Record size, record format, field sizes, and field data formats for DOS files are not available to DOSLINK and must be known by the user in advance. Once a DOS sector is transferred into a TOS string variable, the DOSLINK user is responsible for field and record layouts. The following describe the linkage to DOSLINK.

DOSLINK is parameter driven by the disk's BIOS Parameter Block (BPB). DOSLINK will work if your disk is IBM BIOS compatible, or if the BPB is in standard format, the sector size is 512 bytes and there is a standard IBM Compatible partition boot block on the disk.

IMPORTANT: DOSLINK is designed to run in single-user mode. This is because DOSLINK updates the DOS directory and System tables. If you have more than one user accessing these tables simultaneously they may get garbled, thus destroying the DOS partition. DOSLINK prevents this by returning an Error 18 unless DOSLINK is accessed by the console device (T0).

The format of the call to DOSLINK is:

```
CALL "DOSLINK" ,function (string)
                  ,data (string)
                  ,scratch buffer (string)
                  [,ERR = stmt-no]
```

| | |
|-----------|-------------------------------------|
| function: | "SELECT" = Select DOS disk |
| | "CD" = Select DOS directory |
| | "DIR" = Return directory entry |
| | "OPEN" = Open DOS file |
| | "CREATE" = Create DOS file |
| | "READ" = Read sectors from DOS file |
| | "WRITE" = Write sectors to DOS file |
| | "CLOSE" = Close DOS file |
| | "ERASE" = Erase DOS file |
| | "END" = End of DOSLINK job |

| | |
|------|---|
| data | String which contains data for function. The format of this data is dependent on the function. Each function checks the input data string for size and content. |
|------|---|

If it fails this test, an Error 37 is returned immediately.

scratch buffer: This string is used internally by DOSLINK. It must not be used by the calling program. The same string must be used for each call to DOSLINK.

SELECT - Select DOS Disk

This call sets a new environment for DOSLINK. It must be the first call in the series of calls to access DOSLINK. If you call another DOSLINK function before you call SELECT, DOSLINK will return an Error 36.

The data string must be a 2-character Thoroughbred logical disk number on the physical drive. For instance, if the Thoroughbred disk D0 is on fixed disk H0, DOSLINK will access the DOS partition on the disk. DOS refers to this as disk C. The normal correlation between Thoroughbred and DOS drives is:

| <u>TOS</u> | <u>DOS</u> |
|------------|------------|
| H0 | C: |
| H1 | D: |
| F0 | A: |
| F1 | B: |

Either fixed disk or diskette may be specified. If you specify diskette, the DOS diskette must be mounted before the SELECT function is executed.

IMPORTANT: DOSLINK will automatically DISABLE the Thoroughbred directory on this disk in order to accomplish PUTs of data into the DOS partition. This means that no other users can have a file open in this directory or DOSLINK will return an Error 17. Also, when the directory is disabled, no other users will be able to access data in this directory.

DOSLINK returns an Error 17 when:

- You specify a disk which is not configured.
- There is no DOS partition on the specified drive.
- You specify a diskette which is not a DOS formatted diskette.
- DOSLINK cannot DISABLE the disk.

WARNING: If you modify the scratch buffer string in any way, DOSLINK could wipe out the DOS and/or TOS partition, destroying all data on your hard disk.

If DOSLINK successfully accesses DOS data on the drive, the scratch buffer is initialized. Don't do anything to this string. Keep using it in all your calls to DOSLINK because this is where DOSLINK keeps its internal pointers and sector buffers.

The format of the returned data string is:

| | | |
|-------|-------|---|
| Bytes | 1-2 | Number of bytes per sector (Binary) |
| Byte | 3 | Sectors per cluster (Binary) |
| Bytes | 4-5 | Number of reserved sectors (Binary) |
| Byte | 6 | Number of File Allocation Table (FAT) entries(Binary) |
| Bytes | 7-8 | Maximum number of root directory sectors (Binary) |
| Bytes | 9-10 | Number of sectors in partition (Binary) |
| Byte | 11 | Media Descriptor (HEX) |
| | | \$F8\$ = Fixed disk |
| | | \$F9\$ = High density diskette (1.2 MB) |
| | | \$FD\$ = Double-sided diskette - 9 sector |
| Bytes | 12-13 | Number of sectors in one FAT (Binary) |
| Bytes | 14-15 | Number of sectors per track (Binary) |
| Bytes | 16-17 | Number of heads (Binary) |
| Bytes | 18-21 | Number of hidden sectors (Binary) |

The information in this string comes from the disk's BIOS Parameter Block. The binary fields have been converted into TOS format. When you get control back from the SELECT function, the root directory is selected.

The standard sector size for DOS disks is 512 bytes and the size for TOS is 256 bytes. You must be careful to remember that the READ and WRITE functions use the DOS (512 byte) size in their calculations.

CD - Select Directory

This call selects as the new current directory, a subdirectory of the current directory.

The format of the input data string is:

| | | |
|-------|------|-----------|
| Bytes | 1-8 | File Name |
| Bytes | 9-11 | Extension |

A special case returns to the root directory:

"\"

Note that even though the backslash character used to denote 'root' is only one character long, the entry is padded with spaces to make it 11 characters long.

DOSLINK returns the number of directory entries in the data string in binary format.

| | | |
|-------|-----|---------------------------------------|
| Bytes | 1-2 | Number of directory entries |
| Bytes | 3-4 | Number of available directory entries |

By repeated use of the CD function (and possibly the DIR function), you can walk the directory chain to the directory you desire.

If you specify a subdirectory which does not exist or give a name which is a file rather than a directory, DOSLINK returns an Error 17.

DIR - Return Directory Entry

This call returns an entry of the current DOS directory (see C Function). Each time the DIR function is called, the next entry is returned in the data string. Therefore, you can get all the directory entries by repeating this function. The entries will be returned in the order in which they appear in the DOS directory; DOS does not sort the entries in its directory. When there are no more files, the data string will be null.

The format of the returned string is:

| | | |
|-------|-------|----------------------------------|
| Bytes | 1-8 | File Name |
| Bytes | 9-11 | Extension |
| Byte | 12 | Attribute (Bit Mapped Hex) |
| | | Bit 0 - \$01\$ Read Only |
| | | Bit 1 - \$02\$ Hidden File |
| | | Bit 2 - \$04\$ System File |
| | | Bit 3 - \$08\$ Volume Label |
| | | Bit 4 - \$10\$ Subdirectory |
| | | Bit 5 - \$20\$ Not Archived |
| Bytes | 13-18 | Creation Time (HHMMSS) |
| Bytes | 19-24 | Creation Date (YYMMDD) |
| Bytes | 25-26 | Starting cluster number (Binary) |
| Bytes | 27-30 | File size in bytes (Binary) |

The information in this string comes from the directory entry. The binary fields have been converted into TOS format. The time and date fields have been converted into strings.

DOS requires every directory entry to be 11 characters long. If the entry you want is shorter than 11, it must be padded with space characters to bring it up to 11. Also, if a file extension exists, it must reside in bytes 9 through 11. Note that the period between the File Name and any File Extension is not part of the 11 bytes.

OPEN - Open DOS File

This function selects a DOS file in the current directory. Subsequent I/O function calls may read from or write data to this file. The input data string contains the full name and extension in the format specified below. Each field is left justified, with trailing spaces. If there is no extension, that field will be blank.

The format of the input data string (with no period between File Name and Extension) is:

| | | |
|-------|------|-----------|
| Bytes | 1-8 | File Name |
| Bytes | 9-11 | Extension |

DOSLINK returns the file's directory entry (see DIR function). If you specify a file which does not exist, DOSLINK returns an Error 12.

CREATE - Create DOS File

This function creates a new DOS file in the current directory. Subsequent I/O function calls may read from or write data to this file. The input data string contains the full file name and extension in the format specified below. If there is no extension, that field will be blank.

The format of the input data string (with no period between File Name and Extension) is:

| | | |
|-------|------|-----------|
| Bytes | 1-8 | File Name |
| Bytes | 9-11 | Extension |

If you specify a file which already exists, an Error 12 is returned. If DOSLINK is unable to find space for the file's entry in the current DOS directory, an Error 16 is returned. Note that DOSLINK cannot expand a DOS directory, so there must be an available slot for the file's entry.

READ - Read Sectors from DOS File

This function causes data to be read from the selected DOS file. This is a "random" READ; you can skip around in a DOS file any way you choose.

The input data string contains the following in binary format:

| | | |
|-------|-----|---|
| Bytes | 1-2 | Starting sector number to read. The file's first sector is Ø. |
| Bytes | 3-4 | Number of sectors to read. |

DOSLINK will return the selected sectors in the data string. If you select sectors that go beyond the End of File, the data string will contain only the number of bytes from the start of the first selected sector to the End of File. If you select a starting sector beyond the End of File, the data string will be null.

NOTE: the sector numbers referred to here are relative sector numbers within the file, not physical sector numbers accessed through BASIC

WRITE - Write Sectors to DOS File

This function causes data to be written to the selected DOS file. This is a "random" WRITE; you can skip around in a DOS file any way you choose. Note however, that if you don't write each sector, the file may not be readable by some DOS programs. Remember, the logical End of File is determined by the highest sector you write, even if not all sectors have been written.

The input data string contains the following in binary format:

| | | |
|-------|-----|--|
| Bytes | 1-2 | Starting sector number to write. The file's first sector is Ø. |
| Bytes | 3+ | The data to write. |

Since you must start each write at a sector boundary, the size of your data should be a multiple of 512 (the DOS sector size). An exception is the sector you write with the highest sector number. This sector determines the logical End of File, so if you want the length of your DOS file to be other than a multiple of 512, you can make your data string another length.

The length of the file is determined by the highest sector that is written to it. The file length is always saved when you create a file. It is also saved when you write beyond the End of File on a file that you opened.

NOTE: The sector numbers referred to here are relative sector numbers within the file, not physical sector numbers accessed through BASIC. DOSLINK does

not keep track of the amount of available space in the DOS partition or on the floppy diskette; if an attempt is made to write to DOS when there are no more available sectors, an Error 15 is returned.

CLOSE - End I/O to DOS File

This function ends the communication with the selected DOS file. This function must be called before another file may be accessed through DOSLINK.

If you have changed the file's size and don't call this function, the new data is not saved as part of the file, and you may have caused some "orphan" sectors to be created in the DOS partition. A "dummy" string must be passed in place of the data string parameter to hold its place.

Only one DOS file can be open at one time. If you attempt to call a function other than READ, WRITE, or CLOSE when a file is open, an Error 13 is returned.

If you attempt to call one of the READ, WRITE, or CLOSE functions when no file is open, an Error 14 is returned.

ERASE - Erase DOS File

This function erases a DOS file in the current directory. The format of the input data string is:

| | | |
|-------|------|-----------|
| Bytes | 1-8 | File Name |
| Bytes | 9-11 | Extension |

If you specify a file which does not exist, an Error 12 is returned. An Error 13 is returned if the file is open.

END - End of DOSLINK Job

This call destroys the scratch string, preventing DOSLINK from being accessed by anything other than the SELECT function, and re-enables the selected disk number. A "dummy" string must be passed in place of the date string parameter to hold its place.

Sample Program - Reading DOS File

1000 CALL "DOSLINK", "SELECT", "D0", Z\$

Select the DOS partition on the same disk as Thoroughbred disk D0. On most systems, this will be DOS drive C. DOSLINK is given the Z\$ string. It will be used in all DOSLINK calls, and its value will not change.

1100 CALL "DOSLINK", "DIR", F\$, Z\$

1110 IF F\$(1,11) = "INVOICE " AND F\$(12,1) = "\$10\$ GOTO 1200

1120 IF F\$ < > "" GOTO 1100

1130 PRINT "INVOICE DIRECTORY NOT FOUND"; GOTO 9000

Look for the INVOICE directory in the root directory.

1200 CALL "DOSLINK", "CD", "INVOICE ", Z\$

Select the INVOICE directory.

1300 CALL "DOSLINK", "OPEN", "JAN86 INV", Z\$, ERR = 1320

1310 GOTO 1400

1320 PRINT "INVOICE FILE NOT FOUND"; GOTO 9000

Select the January 1986 Invoice file to read.

1400 OPEN (1)"8601IV"

Open an existing Thoroughbred file. Let's assume this file has 512-byte records. We want to copy the DOS file exactly as it exists.

1500 S = Ø

Keep the current DOS sector number in S.

1600 S\$ = BIN(S,2) + BIN(1,2)

1610 CALL "DOSLINK", "READ", S\$, Z\$

Read the next DOS sector. Notice the way we built S\$.

1700 IF S\$ = "" GOTO 2000; REM "End of File check."

1800 WRITE RECORD (1) S\$

Copy the DOS data to the Thoroughbred file.

1900 S = S + 1

1910 GOTO 1600; REM "End of copy loop."

2000 CALL "DOSLINK", "CLOSE", S\$, Z\$

2010 CALL "DOSLINK", "END", " ", Z\$

Close the DOS file and end the job. Notice that S\$ and " " are "dummy" parameters in these calls.

Sample Program - Writing DOS File

1000 CALL "DOSLINK", "SELECT", "D9", Z\$

Select the DOS partition on the same disk as Thoroughbred disk D9. On most systems, this will be DOS diskette drive A.

DOSLINK is given the Z\$ string. It will be used in all DOSLINK calls, and its value will not change.

1300 CALL "DOSLINK", "CREATE", "JAN86 INV", Z\$, ERR = 1320

1310 GOTO 1400

1320 PRINT "NO DOS DIRECTORY ENTRY AVAILABLE"; GOTO 9000

Create the January 1986 Invoice file in the root directory.

1400 OPEN (1)"8601IV"

Open the Thoroughbred file. Let's assume this file has 512-byte records. We want to copy it to the DOS file exactly as it exists.

1500 S = Ø

Keep the current DOS sector number in S.

1600 READ RECORD (1, END = 2000) S\$

Read the next record from the Thoroughbred file.

1700 S\$ = BIN(S,2) + BIN(1,2) + S\$

1710 CALL "DOSLINK", "WRITE", S\$, Z\$, ERR = 1730

1720 GOTO 1800

1730 PRINT "ERROR WRITING TO DOS DISKETTE"; GOTO 5000

Write the next DOS sector. Notice the way we built S\$.

1800 S = S + 1

1810 GOTO 1600

End of copy loop.

2000 CALL "DOSLINK", "CLOSE", S\$, Z\$

2010 CALL "DOSLINK", "END", S\$, Z\$

Close the DOS file and end the job. Notice that S\$ is a "dummy parameter in these calls.







*Thoroughbred/OS™
Reference Manual Addenda*

Thoroughbred/OS™
Reference Manual Addenda

COPYRIGHT © 1988 CONCEPT OMEGA CORPORATION
All Rights Reserved.



This documentation may not be reproduced or modified in any way, without the express written permission of:

Thoroughbred®
Concept Omega® Corporation

19 Schoolhouse Road • PO Box 6712 • Somerset, NJ 08875-6712
Phone: 201-560-1377 • Telex: 910-3808-394 • Fax: 201-722-7958

#TRMA-051688-MISC

Thoroughbred, Concept Omega, Thoroughbred Business BASIC, Thoroughbred Business BASIC Operating System are trademarks of Concept Omega Corporation.

Thoroughbred/OS™

Reference Manual Addenda

Table of Contents

DIRECTIVES

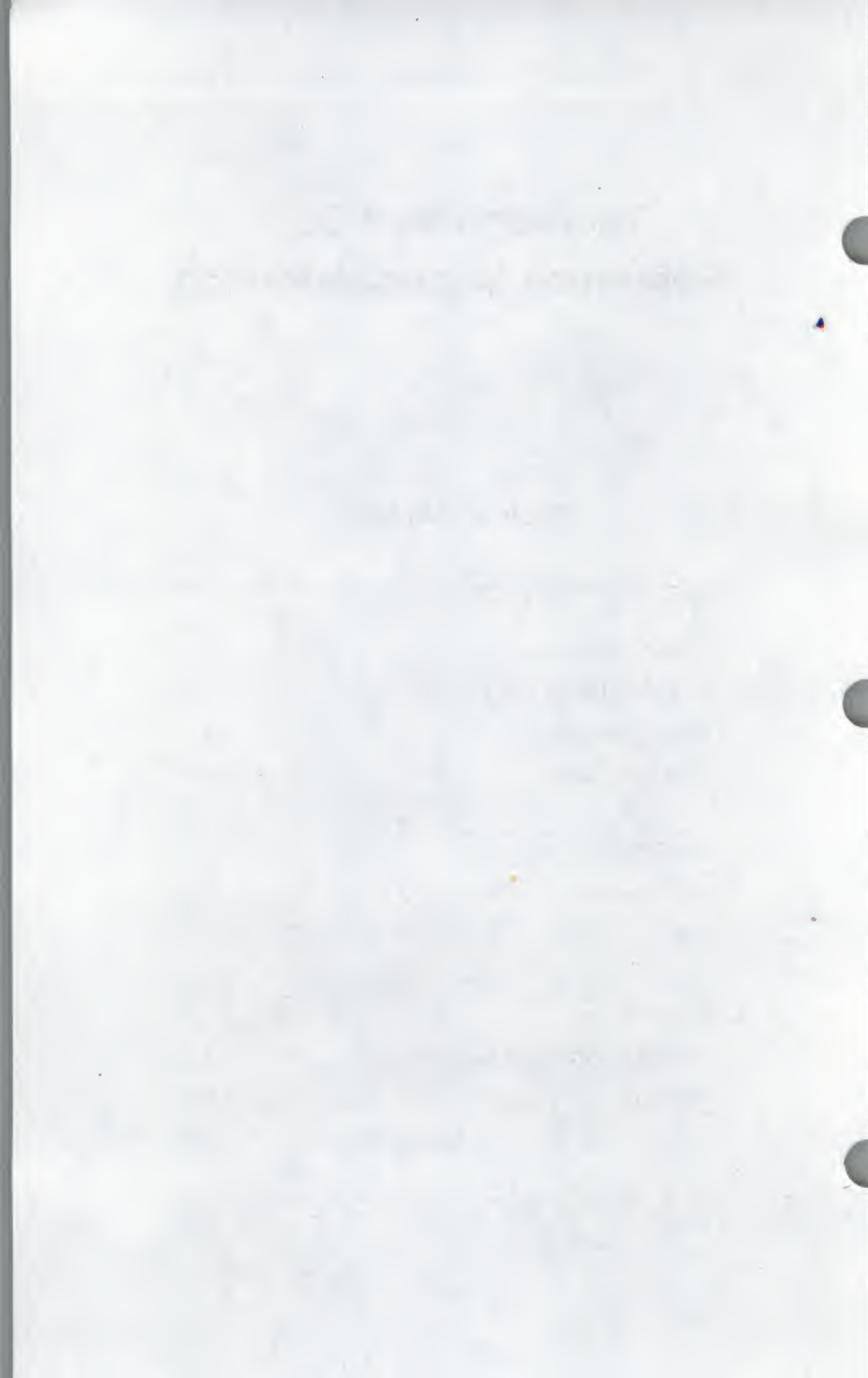
| | |
|-------------------|---|
| SERIAL Directive | 1 |
| ENCRYPT Directive | 2 |
| PSAVE Directive | 3 |
| LOAD Directive | 4 |

FUNCTIONS

| | |
|--------------|---|
| DSD Function | 5 |
| CVT Function | 6 |

UTILITIES

| | |
|---------------------------------------|----|
| *NPSD BASIC Configurator Utility | 7 |
| *UPSD Switch Floppy/Hard Disk Utility | 23 |
| *RECOVER Bad Track Sparing Utility | 27 |



Thoroughbred/OS Manual Addenda

SERIAL Directive

This directive defines a Serial file on disk. The following is the syntax for the SERIAL directive:

SERIAL file-id, no-recs, rec-sz, disk-no, sctr-no [,ERR = stmt-no]

Where:

| | |
|-----------------|---|
| file-id | A 1 to 8 character file name. |
| no-recs, rec-sz | Positive integers used to calculate the file size. |
| disk-no | The logical disk on which to locate the file. |
| sctr-no | An integer specifying the starting sector number for locating the file. |
| stmt-no | The number of the statement to branch to if this directive produces an error condition in RUN Mode. |

Specifying a sector number of 0 directs the system to handle the assignment of storage location.

Serial files contain variable-length records that can only be accessed sequentially. Record size is limited only by the maximum amount of data that can be processed in an I/O operation up to 32K. Serial files are typically used for spooling operations.

Index values may be used to READ a Serial file. If an IND other than the next higher sequential record is specified, BASIC will access one record at a time, working in either a forward or reverse direction.

A Serial file must be LOCKed in order to WRITE to it. All records written to a Serial file are appended to the end of the file unless the file is erased first. If Index values are used to WRITE to a Serial file, BASIC will ignore them.

ENCRYPT Directive

NOTE: Encrypted programs will not LOAD or RUN on releases of BASIC prior to Version 7.2 and TOS Version 6.6.

This directive encrypts or decrypts a program file. Encryption does not affect the executability or normal functioning of a program, and acts as a safeguard without impairing program operation or useability. Following is the syntax for the ENCRYPT directive:

```
ENCRYPT "src-prog-id", "trgt-prog-id" [,PWD = "pswd"]  
      [,ERR = stmt-no]
```

Where:

| | |
|--------------|---|
| src-prog-id | Specifies the name of an existing program to be encrypted or decrypted. |
| trgt-prog-id | Specifies a target program which will become the newly encrypted or decrypted file. |
| pswd | Specifies an optional password from 4 to 8 characters in length. |
| stmt-no | Specifies the statement to branch to if an error occurs in processing this directive. |

There is little restriction on encrypting a program. When a standard, un-encrypted program is specified as the source, the target program will receive an encrypted copy of the source.

You can specify the encrypted program to be either reversible or irreversible. Reversible programs can be decrypted while irreversible programs cannot be decrypted. To create a reversible, encrypted program you **MUST** specify a password (PWD = option) in the ENCRYPT statement. To create an irreversible, encrypted program, no password is specified in the ENCRYPT statement: either no PWD = option or PWD = with a null password (PWD = ""). If a null password is specified, the system automatically assigns a random password (practically irreversible).

When a reversible, encrypted program is specified as the source along with the password that was used when encrypting the program, the target program will receive a decrypted copy of the source.

The target program can be (1) an existing program, which will be overwritten, (2) the source program, which will also be overwritten, or (3) an undefined program, which will be created on the same disk as the source program.

WARNING: Beware of creating an encrypted program without a password (irreversible) while specifying the program as both the source and target, because the source program will be overwritten and the procedure cannot be reversed.

Encryption may create a temporary file on disk having a name built from the constant "*EN" plus the Task ID (T0, T1, etc.).

For encrypted programs, the first byte displayed in the *1PSD Utility (27th in file) is \$80\$; for standard, unencrypted programs, it is \$00\$.

The display and modification of an encrypted program is restricted through the following directives: LIST, EDIT, PGM, MERGE, SAVE. An attempt to use these directives on an encrypted program will cause an Error 18 (Secure File Access); however, the LIST directive can still operate on statements 1 through 99. Encrypted program security is complete for a length of 1K starting at statement 100. These restrictions can be temporarily suspended when a reversible, encrypted program is loaded using the PWD = option.

The PSAVE command can also encrypt a program, but it cannot decrypt a program.

PSAVE Directive

This directive saves a program in encrypted format (Protected SAVE) by combining the functions of the SAVE directive with the encrypting capability of the ENCRYPT directive. Following is the syntax for the PSAVE directive:

```
PSAVE ["prog-id" [,prog-sz, disk-no, sec-no]] [,ERR = stmt-no]  
      [,PWD = "pswd"]
```

NOTE: When the ERR or PWD option is not preceded by another parameter, the comma is not used.

PSAVE can operate on programs in the User Task area that are modifiable (standard, unencrypted programs, and reversible, encrypted programs that were loaded with the PWD = option).

If PSAVE operates on an unencrypted program, the statement must specify the PWD = option. If a password is specified, it creates a reversible encrypted program; if a null password is specified (PWD = "") the system automatically assigns a random password (practically irreversible).

If an encrypted program was loaded with PWD = (allowing it to be modified), a PSAVE of this program does not need to specify the PWD = option, which will default to the correct password. PSAVE cannot operate on an encrypted program unless it was loaded with the PWD = option.

The PSAVE command cannot decrypt a program.

Refer to the SAVE directive for an explanation of the program definition features and saving to an existing program file.

LOAD Directive

Enhancement

The PWD = option has been added to the LOAD directive:

LOAD program-id [,PWD = "pswd"]

The PWD = option specifies the password of a reversible, encrypted program so that the program, when loaded, can be modified and displayed. This temporarily removes the restrictions of an encrypted program.

DSD Function

Enhancements

A. DSD("H#")

The DSD function now returns the hard disk parameters if the disk is IBM ROM-compatible.

DSD("H#"), where H# is the hard disk, returns the following information:

| <u>Byte</u> | <u>Description</u> |
|-------------|-----------------------------------|
| 1-2 | Device Designator (H0 or H1) |
| 3 | Status byte (undefined) |
| 4 | Device Type: H |
| 5-6 | Unit #: 60 or 61 |
| 7 | Number of heads (hex) |
| 8-9 | Number of cylinders (hex) |
| 10 | Number of sectors per track (hex) |
| 11-20 | Unused |

*SETDSK or Option 1 of the Installation Menu uses this function to determine the disk parameters if the disk is IBM ROM-compatible.

B. DSD("Tx")

Bytes 19 and 20 of the string returned in the DSD function of the format DSD("Tx"), where Tx is a valid, configured terminal, returns the number of characters in the type-ahead buffer of the specified terminal. (The terminal does not have to be open on a channel for the function to operate.) The value returned can be converted by the DEC function and used in a READ RECORD statement with the SIZ = option to get the actual characters out of the type-ahead buffer.

Example:

```
10 A$ = DSD("T1")
20 A = DEC(A$(19,2))
30 OPEN(1)"T1"
40 READRECORD(1,SIZ = A)I$
```

NOTE: DSD is not supported as a standard part of the language and is subject to change without notice.

CVT Function

A new system function (CVT) has been added to BASIC to provide conversion capabilities for alphanumeric variables. Following is the syntax of the CVT BASIC function:

CVT(string-value,option)

Where:

option An integer value which designates the conversion operations.
(see the following chart.)

| <u>option</u> | <u>hex*</u> | <u>Operation</u> |
|---------------|-------------|--|
| 32 | (\$20\$) | Convert lowercase characters to uppercase. |
| 128 | (\$80\$) | Discard trailing spaces and tabs. |
| 4096 | (\$1000\$) | Convert uppercase characters to lowercase. This has priority over option 32. |

*Hex values shown for bit positioning only.

Operations are additive; for example, n = 160 will perform the operations of both n = 32 (convert lower case characters to uppercase) and n = 128 (discard trailing spaces and tabs).

*NPSD - BASIC Configurator Utility

The *NPSD Utility - BASIC Configurator has been revised to include a Terminal Configurator in addition to the other selections. The Terminal Tables are now available as Selection 6 on the Configuration Type menu. The following sample Configuration Type menu reflects the new selection.

```

** THOROUGHbred/OS BASIC CONFIGURATOR-VERSION **--DATE **
THE FOLLOWING TYPES OF SYSTEM INFORMATION ARE CONFIGURABLE:

    1. SERIAL PORTS
    2. DISK DIRECTORIES
    3. MNEMONIC TABLES
    4. SYSTEM OPTIONS
    5. MEMORY BANKS
    6. TERMINAL TABLES

ENTER 'X' TO CANCEL CONFIGURATION JOB
    'Y' TO END JOB NORMALLY (SAVE NEW CONFIGURATION)

NOTE: 'X' AND 'Y' OPTIONS ONLY AFFECT THOSE CHANGES MADE WITH
      SELECTIONS 1 THROUGH 5. TERMINAL TABLE CHANGES (MADE WITH
      SELECTION 6) ARE SAVED AS THEY ARE MADE.

OS TABLE SIZE: *****
ENTER NUMBER OF CONFIGURATION TYPE:
```

NOTE: The X and Y options on this menu, used to cancel or save a configuration job, are only applicable to Selections 1 through 5. (The Terminal Configurator, Selection 6, is an independent configuration job that contains options to save or cancel terminal table modifications.)

Due to the new method of configuring terminals using Selection 6, some changes have been made to the existing configuration selections. The changes are as follows:

Selection 1: Serial Ports

The screen and procedure used for configuring serial ports has been modified. A sample Serial Port Configuration Screen is displayed below followed by a listing of the procedure prompts:

| SERIAL PORT CONFIGURATION | | | | | | | | | |
|---------------------------|--------|-------|-------|---------------|----------------|------|---------|---------------------|------------------|
| PORT | DEVICE | MODEL | BAUD | PRO- TICAL | AUTO- START | BANK | PROGRAM | ATTACHED PRINTER | PRINTER MODEL |
| 0 | T0 | ---- | ***** | | Y | 1 | **PSD | | |
| 1 | T1 | ---- | 19200 | E71S | Y | 6 | **PSD | P2 | 0300 |
| 2 | T2 | ---- | 09600 | E71S | Y | 2 | **PSD | P3 | 0300 |
| 3 | T3 | ---- | 09600 | E71S | N | | | | |
| 4 | T4 | ---- | 19200 | E71S | Y | 4 | **PSD | | |
| 5 | T5 | ---- | 09600 | E71S | Y | 5 | **PSD | | |
| 6 | P1 | ---- | 09600 | N81S | - | | | | |

ENTER PORT NUMBER TO MODIFY <N>ext page, <P>revious page, or <CR>
to exit:

NOTE: This screen shows ports 0 through 5 configured as terminal "T0" through "T5". (Terminal "T0" is the console or monitor.) Port 6 is configured as printer "P1".

ENTER PORT NUMBER TO MODIFY < N > ext page, < P > revious page,
or < CR > to exit:

Enter the port number of the terminal or printer you want to configure. It is recommended that you enter a < Return > in response to the following prompts (where applicable) to accept the default value presented:

DELETE ENTRY? (Y/C.R.)

ESCAPE KEY WILL CANCEL CHANGES

RETURN KEY WITH NO INPUT WILL GO TO NEXT FIELD

DEVICE CODES ARE T0-TZ (TERMINALS); P1-PZ (PRINTERS)

ENTER DEVICE CODE:

Valid "device codes" for terminals are T0 through T9 and TA through TZ. Valid "device codes" for printers are P1 through P9 and PA through PZ. You cannot select a device code that is already configured.

ENTER COMMUNICATION PROTOCOL PARAMETERS:
PARITY (E FOR EVEN, O FOR ODD, N FOR NO PARITY):

CHARACTER LENGTH (7 OR 8 BITS PER CHARACTER):

NUMBER OF STOP BITS (1 OR 2 BITS):

PROTOCOL (S FOR XON/XOFF, H FOR DTR, B FOR BOTH, N FOR NONE):

ENTER BAUD RATE:

Enter the valid baud rate for your terminal (150, 300, 600, 1200, 2400, 4800, 9600, or 19200).

AUTO START (Y/N):

ENTER BANK NUMBER:

ENTER PROGRAM (BLANK FOR CONSOLE MODE):

This specifies the name of a program to run at the terminal each time the system is started.

ATTACHED PRINTER (Y/N):

If you enter Y the system prompts you for the printer device code and printer model code:

ENTER ATTACHED PRINTER DEVICE CODE (P1-PZ):

Valid "device codes" for serial printers are P1 through PZ. You cannot select a printer name that is already configured.

ENTER ATTACHED PRINTER MODEL CODE:

Valid model codes are 0300 and 1650. The model code you use depends upon whether or not your printer is configured for auto line feed.

Selection 3: Mnemonic Tables

The Mnemonic Tables are no longer applicable to terminals; under this selection, Mnemonic Tables are now configured only for printers. The following Mnemonic Tables may be available.

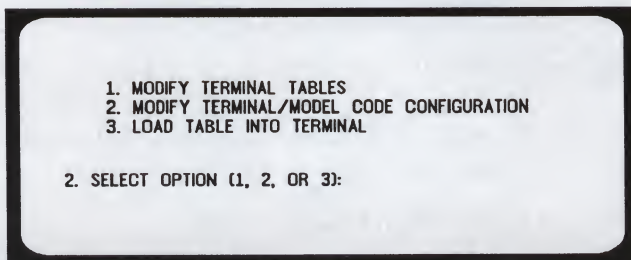
| | |
|------|--------------------|
| 0300 | Printronix Printer |
| 1650 | Diablo Printer |

Selection 6: Terminal Tables

Thoroughbred BASIC can only function with a terminal that has the ability to roll the screen up one line when data is entered on the last line of the screen.

It may be necessary to have the terminal manual in order to obtain detailed information concerning all terminal functions and the proper codes and escape sequences acceptable to the terminal.

Selection 6 allows you to define and configure terminal tables for the terminals on your system. When you enter selection 6, the system will display the Terminal Configurator screen. The following is a sample screen.



Option 1, Modify Terminal Tables, will display a list of existing terminal tables. This option will allow you to define or modify a terminal table containing terminal characteristics and mnemonics that will be used by BASIC to control the terminal.

TOS is shipped with many terminal tables already predefined, and in most cases, one of these tables can be used making it unnecessary to define a new table. Therefore, you should first check this DEFINED TERMINAL TABLE LIST under Option 1 for your terminal type. If your terminal is not listed, you can create a new terminal table.

Option 2, Modify Terminal/Model Code Configuration, will allow you to associate a terminal table with a terminal ID.

Option 3 is not supported in this version. However, the same operation is accomplished when the system is rebooted; at that time the terminal tables specified in Option 2 will be loaded for the terminals in order to activate their characteristics. The terminal tables are maintained in the Direct File named TCONFIG (TOS is distributed with a TCONFIG file.) If this file cannot be opened by BASIC (e.g., TCONFIG has been deleted, or is located on a disabled directory), the configurator will attempt to create a new TCONFIG. The new TCONFIG will be created on the UTIL directory (D0). If UTIL cannot be accessed, the system will prompt you for a disk directory from 0 to 9 where TCONFIG can be created:

ENTER DISK NUMBER WHERE 'TCONFIG' CAN BE CREATED

If the file cannot be created, the system responds with:

CANNOT FIND OR CREATE 'TCONFIG' FILE

CONFIGURATOR TERMINATED ABNORMALLY

The system will return to BASIC Console Mode. If this happens, try to recreate TCONFIG from a backup or the master copy of TOS.

All prompts in the Terminal Configurator are numbered. Option 1 begins with Prompt 3; Option 2 with Prompt 22. The following keywords are among the valid responses:

| | |
|-------|--|
| END | Allows you to exit the configurator and save or cancel the modified or newly defined terminal table (Prompts 20 and 21). |
| BACKn | Causes the configurator to return to the preceding prompt or to the prompt number specified by n. |
| TABLE | Causes the configurator to redisplay the terminal table list, or under Option 2, the Terminal/Model Code table. If you have proceeded beyond Prompt 7, an option will be given to save the terminal table that is being created or modified before reprinting the terminal table list. |
| NULL | Causes the configurator to generate a null entry for a give prompt. |

< Return > A Carriage Return will cause the configurator to retain the existing value for a given parameter.

1. Modifying Terminal Tables

Select Option 1 on the Terminal Configurator screen. The system will display a list of predefined terminal types. The following is a sample screen and may vary from the list provided with your system:

| DEFINED TERMINAL TABLE LIST | | | |
|---|-----------------------|---------|--------------------|
| ADM-3A | LSI MODEL ADM-3A | SEIKO | SEIKO 880 TERM |
| ALP62A | Alpha Micro 62-A | TVI912 | TELEVIDEO 912 |
| ALTOS-II | ALTOS-II TERMINAL | TVI920 | TELEVIDEO 920 |
| AMPEX | AMPEX TERMINAL | TVI950 | TELEVIDEO 950 |
| ANNARBOR | AMBASSADOR | VT100 | DEC VT100 |
| B4 7250 | BASIC/FOUR 7250 | VT52 | DEC VT52 |
| CIT-80 | C ITOH MODEL 80 | WY-50 | WYSE WY-50 DISPLAY |
| COMM | COMMUNICATIONS PORT | WY-60 | WYSE 60 |
| DM 10 | DATAMEDIA COLOR 10 | WY50MGC | WYSE 50 (SPECIAL) |
| HZ1510 | HAZELTINE 1510 | WYSE75 | WYSE 75 |
| IBM PC | IBM MEMORY MAPPED | Z19 | ZENITH TERMINAL |
| IBM3101 | IBM 3101 | ZDS | HEATH TERMINAL |
| KIMKT7 | Kimtron KT-7 | ZEN8001 | ZENTEC MODEL 8001 |
| MICOTERM | MICRO TERM | ZEPHYR | ZENTEC ZEPHYR |
| REMOTE | REMOTE MEMORY WRAPPED | | |
| 3. C-CREATE M-MODIFY D-DELETE L-LIST TABLE ENTER C, M, D, OR L: | | | |

Each terminal table is identified by a Model Code. These Model Codes are displayed on the Defined Terminal Table List along with the corresponding descriptions of the terminals. When you Create, Modify Delete, or List a table, the system will display the following prompt and you will enter the Model Code of a new or existing table:

4. ENTER TERMINAL TABLE:

List Table on Printer

At Prompt 3, enter L in order to List a Terminal Table. This option will allow you to print either the Defined Terminal Table List or selected terminal table. The following prompt will appear:

TABLE LISTING ONLY (Y/N)?

This refers to the Defined Terminal Table List displayed on the screen. A Y response will allow you to select a printer on which to print the Defined Terminal Table List. An N response will allow you to select a specific terminal table

to print. If you attempt to print a nonexistent table, the system will display an INVALID message.

Delete a Table

At Prompt 3, enter D in order to Delete a terminal table. The system will prompt for a terminal table. If you enter a nonexistent table, the system will display an INVALID message. Otherwise, the system will respond with:

4.1 DELETE (Y/N)?

WARNING: When you enter Y to execute the delete, the table will be immediately deleted from the system without providing you the opportunity to cancel the operation.

Create a New Table

At Prompt 3, enter C in order to Create a new terminal table. The system will prompt you for a terminal table, and you must enter a new Model Code from 1 to 8 characters in length that will be used to identify the table. (If you enter a Model Code that already exists, the system will display an INVALID message.) The system will allow you to base the new terminal table upon an existing table or to start from scratch; the system will prompt with:

5. ENTER BASE TERMINAL TABLE OR CARRIAGE RETURN:

At this point, you can enter the Model Code of an existing terminal table and the system will use the parameters of that table as default values. If you enter a <Return>, the system will allow you to create the table from scratch.

The system will then prompt you for each value to be entered into the new table. These prompts will be the same as those described under the option to Modify a Table (proceed to Prompt 6).

Modify a Table

At Prompt 3, enter M in order to Modify an existing terminal table. After you enter the Model Code of an existing terminal table, the system will display a series of prompts (described following) and allow you to enter new values or <Return> to retain the existing values.

6. ENTER TERMINAL NAME (20 CHARACTERS MAX)

Enter a new name or a <Return> to retain the previously defined terminal name.

7. ENTER TERMINAL BACKSPACE CODE (HH)

Enter a hex value or a <Return>. Since most terminals use a hex value of 08 for backspace, the configurator will have the hex value of 08 as default.

8. ENTER TERMINAL NULL CHARACTER CODE (HH)

Enter a hex value or a <Return>. Since most terminals use a hex value of 00 for null, the configurator will have the hex value of 00 as the default.

9. ENTER BASIC ESCAPE CODE (HH)

Enter a hex code or a <Return>. Since most users desire 1B as the escape code that BASIC will recognize, the configurator will supply the hex value of 1B as a default. This should not be confused with Prompt 1 which is the escape code that is transmitted to the terminal to invoke terminal special features.

10. ENTER TERMINAL ESCAPE CODE (HH)

Enter a hex value or a <Return>. Since most terminals recognize a hex value of 1B for an escape, the configurator will supply the hex value of 1 as a default. This should not be confused with Prompt 9 which is the escape code BASIC will recognize from a terminal.

This hex code is used in conjunction with Prompt 11 to provide keyboard entry for the escape code that is to be transmitted to the terminal as part of the mnemonics (described later).

11. ENTER A CHARACTER THAT REPRESENTS AN ESCAPE (X)

Enter one character or a <Return>. Since the definition of terminal mnemonics can require an escape code followed by some character(s), this character will be used to syntactically represent the escape code specified by Prompt 10. A <Return> will default to the character E.

12. ENTER CODE(S) THAT ARE TO BE IGNORED

Enter one or more hex codes or a <Return>. Since some terminals transmit codes back to the CPU after the completion of certain terminal functions, it may be necessary to have BASIC ignore these codes since the application program

might not know what to do with them. A <Return> will indicate that there are no codes to be ignored.

A sample input of 0102FEFF would indicate that the four hex codes 01 02, FE, and FF are to be ignored when these codes are transmitted from the terminal to BASIC. Take care when specifying these codes to ensure that valid keyboard character codes are not ignored.

13. ENTER LEADING BLANK REPLACEMENT CHARACTER (HH)

Enter a hex code or a <Return>. Some terminals require a special code to go in place of the first leading blank that is transmitted to the terminal. If the code is not supplied, leading blanks may be lost.

14. ENTER THE NUMBER OF TERMINAL FUNCTION KEYS:

Enter a numeric value from 4 to 99 or a <Return>. A <Return> will default to 4. After specifying the number of function keys, the configurator will prompt for the hex values that the terminal transmits when a function key is entered:



```
14.1 ENTER FUNCTION KEY 01 CODE
14.2 ENTER FUNCTION KEY 02 CODE
14.3 ENTER FUNCTION KEY 03 CODE
14.4 ENTER FUNCTION KEY 04 CODE
14.n ENTER FUNCTION KEY n CODE
```

Enter a hex value that will be transmitted from the terminal to the CP for each function key. The hex value can be one or more hex codes Examples: 01, 02, 03, 04, 0F, 01400D, 01410D, 01420D, 01430D, 014F0D Consult your terminal manual to determine whether one, two, or more hex digits are needed for each function key value.

It is important to note that hex values can be specified for terminals that do not have function keys. In this case, the codes recognized as function keys are also specified.

15. DOES TERMINAL HAVE CLEAR FOREGROUND (Y/N)?

Enter Y if the terminal has the 'CF' mnemonic; enter N if the terminal does not. This means that BASIC must keep a bit map for data that is printed in the foreground so that the 'CF' (Clear Foreground) mnemonic can clear the foreground data using the bit map.

16. ENTER CURSOR CONTROL FORMAT MASK (APYLZ)

This parameter is used for cursor positioning functions in BASIC, e.g., PRINT @(10,15), "REPORT #1".

Examples: AAPL AALP AAAPYLZZ APL NULL

- A represents a leading control character
- P represents a cursor position within a line value
- Y represents a control character to follow the cursor control position value
- L represents a cursor line value
- Z represents a control character to follow the cursor control line value

As shown in the examples, a mask must be entered that represents the format of the cursor control sequence. Once the mask is defined, the system prompts for specific hex values to be used for each part of the mask. Prompts are issued in the following order: A's, P, Y's, L, Z's.

Prompts are only issued for characters that are in the mask. More than one A, Y, or Z may be in the mask, indicating more than one character is to be transmitted for each code. Multiple occurrences of the characters A, Y, and Z must be contiguous. That is, AAPL is valid and ALPA is not. Both the L and P codes are required and can occur only once.

A sample input of the mask AALP would result in the following prompts (sample operator responses are shown):

```
16.1 VALUE FOR AA (HH): 1B01
16.2 STARTING VALUE FOR P (HH): 20
16.3 NUMBER OF CHARS PER LINE FOR P: 80
16.4 STARTING VALUE FOR L (HH): 20
16.5 NUMBER OF LINES FOR L: 24
```


In the preceding example, the cursor control sequence will consist of a hex 1B01, followed by the cursor position number with the starting value for position 0 (space-based), followed by the cursor line number with the starting value for line 0 (space-based). (The starting value for P and L will be the base value: zero-based (hex 00), one-based (hex 01), space-based (hex 20) or any other base that is required by the terminal.)

Entering the mask AAPYLZZ would result in the following:

```
16.1 VALUE FOR AAA (HH): 1B0102
16.2 STARTING VALUE FOR P (HH): 00
16.3 NUMBER OF CHARS PER LINE FOR P: 80
16.4 VALUE FOR Y (HH): 3B
16.5 STARTING VALUE FOR L (HH): 20
16.6 NUMBER OF LINES FOR L: 24
16.7 VALUE FOR ZZ (HH): 6667
```

In the example above, the cursor control sequence will consist of a hex 1B0102, followed by the cursor position number with the starting value for position 0 (hex 00), followed by a hex value 3B to terminate the line position, followed by the cursor line number with the starting value for line 0 (hex 00), followed by a hex value of 6667 to terminate the line value. This cursor control format is by far the most cumbersome. Fortunately, most terminals do not require this format; most terminals follow the pattern AAPL.

A NULL mask can be used to disable the cursor positioning functions. In this case, any cursor positioning commands (PRINT @. . .) in BASIC will be ignored.

17. BINARY FORMAT FOR CURSOR POSITION AND LINE (Y/N)

If Y is entered, the configurator will expect a one-byte binary value for the cursor position and line values. The valid range for cursor position will be from the hex starting value specified for P, plus the value specified for the number of characters per line. The valid range for line position will be from the hex starting value specified for L, plus the number of lines.

If N is entered, the configurator will assume that the cursor position and line values will be in ASCII format: one byte for each digit, cursor position, or cursor line number.

Examples:

| <u>POSITION</u> | <u>ASCII</u> |
|---------------------|--------------|
| Cursor position 3 | 33 |
| Cursor line 9 | 39 |
| Cursor position 52 | 3532 |
| Cursor line 12 | 3132 |
| Cursor position 123 | 313233 |
| Cursor line 24 | 3234 |

Normally, if a terminal requires this format for the cursor position and line values, the terminal will also require the terminating character (characters Y and Z in cursor mask) for position and line, since the cursor position and line values are variable length, as shown in the above examples.

18. ENTER NUMBER OF NULLS FOR SCREEN ROLL (1 THROUGH 99)

Enter a numeric value from 1 to 99 or a <Return>. Most terminals require a delay following terminal functions that cause the screen to roll. BASIC performs such delays by sending null characters after screen roll functions. This ensures that other valid data sent to the terminal after a screen roll will not be lost.

The system will display a List of Defined Mnemonics, similar to the following screen:

LIST OF DEFINED MNEMONICS

| | | | | | | | | | | | | |
|----|----|----|----|----|----|----|----|----|----|----|----|----|
| VT | EP | FF | BB | EB | BU | EU | SI | SO | BH | EH | BS | |
| CH | CR | LF | ES | RB | CF | CL | CS | LD | LI | TI | SF | SB |

19. ENTER MNEMONIC CODE MM

Enter a new or existing mnemonic code, the keyword TABLE or a <Return>. If a mnemonic code is entered that was previously defined, the option will be given to modify old values that were used to define the mnemonic. If a mnemonic code is entered that was not previously defined, you can enter a new mnemonic definition. If the keyword TABLE is entered, the list of defined mnemonics will be redisplayed.

If a <Return> is entered, the configurator will display the next defined mnemonic and the option will be given to modify the mnemonic.

The following parameters are used to define or modify each mnemonic:

19.1 ENTER CODE TO BE TRANSMITTED TO TERMINAL

Enter NULL, DELETE, a hex value, or a keyboard string value that is to be transmitted to the terminal when the mnemonic is used.

The special code NULL indicates that no codes are to be transmitted, which would cause a mnemonic to be ignored (such as FF, form feed). The special code DELETE indicates that the mnemonic code is to be deleted. The following are examples of valid input:

\$1B31 \$01 \$1BFE91 KE1 KEr KE.0 NULL DELETE

The above examples illustrate that a \$ or K must precede a character string. Strings that are preceded by a \$ are assumed to be hex values. Strings preceded by a K are assumed to be keyboard characters. Note that in the above example, the E is assumed to be an escape code previously defined in Prompt 11.

19.2 ENTER NUMBER OF NULLS TO TRANSMIT AFTER CODE

Enter 0 to 99 or a <Return>. A <Return> will assume that no nulls are to be transmitted after the mnemonic code. Some terminals require a delay after certain terminal commands; transmitting nulls accomplishes this delay.

19.3 ENTER CURSOR UPDATE MODE

Enter one of the following codes that define the position of the cursor after a terminal function is executed:

- 0 This mnemonic does not affect the cursor position.
- 1 The cursor will move to the beginning of the next line. If the function was executed on the last line of the screen, the screen will roll.
- 2 The cursor will move down one line. If the function was executed on the last line of the screen, the screen will roll.
- 3 The cursor will move up one line. If the cursor is on the first line, this function does not move the cursor.
- 4 The cursor will move back one position. If the function was executed on the first character of a line, the cursor will move to the last character.

- ter of the previous line. This function has no effect if executed on the first character of the first line.
- 5 The cursor will move to the beginning of the next line. If this function is executed on the last line of the screen, the cursor will move to the first character of the first line.
 - 6 The cursor will move down one line. If this function is executed on the last line of the screen, the cursor will move to the first line.
 - 7 The cursor will move up one line. If this function is executed on the first line on the screen, the cursor will move to the last line on the screen.
 - 8 The cursor will move back one position. If the function is executed on the first character of a line, the cursor will move to the last character of the previous line. If the cursor is on the first character of the first line, the cursor will move to the last character of the last line.
 - 9 The cursor will move to the first position of the current line.
 - A The cursor will move back one position. If the cursor is on the first position of a line, the cursor will not move.
 - B The cursor will move to the first character of the first line.

19.4 ENTER SPECIAL EDIT CODE

If the terminal does not have a clear foreground capability, specify one of the following codes depending on the type of mnemonic being defined.

- 0 No Special Edit Required
- 1 Clear Screen
- 2 Clear Line
- 3 Line Delete
- 4 Line Insert
- 5 Roll Screen Up
- 6 Clear Foreground
- 7 Roll Screen Down

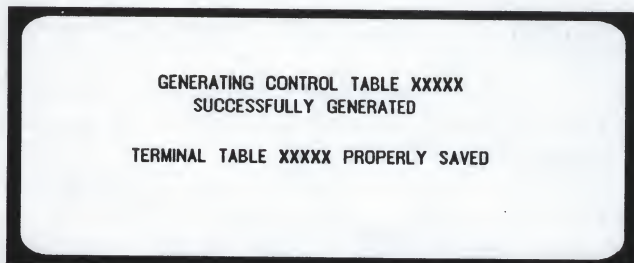
19.5 ENTER FOREGROUND/BACKGROUND CODE

Enter one of the following values or a <Return>. These codes describe how the mnemonic affects the foreground/background status of the terminal.

- 0 Does not affect foreground/background status
- 1 Puts the terminal in foreground mode
- 2 Puts the terminal in background mode

20. SAVE NEW TERMINAL TABLE (Y/N)

If N is entered, the configurator will not save the newly created or modified terminal table. If Y is entered, the configurator will save the newly created or modified terminal table and will display the message:



The system will then provide the option to return to the Terminal Configurator screen to define or modify more tables:

21. CREATE OR MODIFY MORE TERMINAL TABLES (Y/N)

If Y is entered, the system will return to the Terminal Configurator screen (Prompt 2). If N is entered, the Terminal Configurator will terminate and the *NPSD CONFIGURATION TYPE menu will be displayed.

2. Modify Terminal/Model Code Configuration

Selection Option 2 on the Terminal Configurator screen. The system will display a list of terminal ID's and the Model Codes (Terminal Tables) associated with them. The following is a sample screen:

| TERMINAL/MODEL CODES | | |
|----------------------|--------|---------------------|
| T0 | IBM PC | IBM MEMORY MAPPED |
| T1 | COMM | COMMUNICATIONS PORT |
| T2 | TVI950 | TELEVIDEO 950 |
| T3 | WY-50 | WYSE WY-50 DISPLAY |

22. ENTER TERMINAL ID:

The prompt displayed refers to a BASIC terminal ID. Valid ID's consist of the letter "T" followed by any character. If you are defining a new terminal ID (one not displayed), the system will skip the next prompt. Otherwise, if you selected an existing terminal ID, the system will respond with:

23. DELETE (Y/N)

If Y is entered, the entry for the specified terminal ID will be deleted and Prompt 22 will be displayed. If N is entered, the system will allow you to change the Model Code associated with that terminal.

24. ENTER MODEL CODE (8 CHARACTERS MAX)

If the Model Code that you enter does not identify a defined terminal table, the following message will be displayed:

MODEL CODE NOT DEFINED . . . ACCEPTED ANYWAY

The Terminal ID/Model Code association will be entered in the table and the system will redisplay Prompt 22.

Enter TABLE to redisplay the list. Enter END or BACK to exit this selection. If you modified the Terminal ID/Model Code table, the system will display the following prompt and allow you to save or cancel the changes.

SAVE NEW TERMINAL/MODEL TABLE (Y/N)

A Y response will save the new table; an N response will cancel it. The system will return to the Terminal Configurator menu.

NOTE: It is necessary to reboot the system after exiting the *NPSD Utility in order to activate the terminal tables for the terminal specified in Option 2.

*UPSD - Switch Floppy/Hard Disk Utility

This utility is used to switch a removable hard disk or floppy diskette on your system (i.e., to logically attach or detach a floppy disk or removable hard disk). For removable hard disks that have previously been initialized with a Bad Track Table using the *1PSD Utility, this utility will load the table for the disk mounted.

To invoke this utility, enter: ***UPSD <Return>**

This utility may also be invoked as ***U** or as the function number indicated on the Utilities Menu. The system will generate the ***UPSD** screen, similar to the following:

```
*UPSD - SWITCH FLOPPY/HARD DISK

      DISC      DIRECTORY
0   H0         RMOV
1   H1         UTIL
2   H1         PROG
7   H0         HD60
9   F0         TOS

ENTER: 'F' OR 'CR' TO SWITCH FLOPPY DISKETTE OR
      'R' TO SWITCH REMOVABLE HARD DISK

'D' = DISABLED   'R' = RESERVED BY ANOTHER TASK
'L' = LOCALLY DISABLED  'M' = RESERVED BY MY TASK
```

An **<F4>** response will exit to the Utilities Menu.

If you select **R** in order to switch the removable hard disk, the system responds with:

ENTER THE UNIT NUMBER OF YOUR REMOVABLE HARD DISK
(H0 OR H1)

At this point, a **<Return>** will return to the ***UPSD** screen.

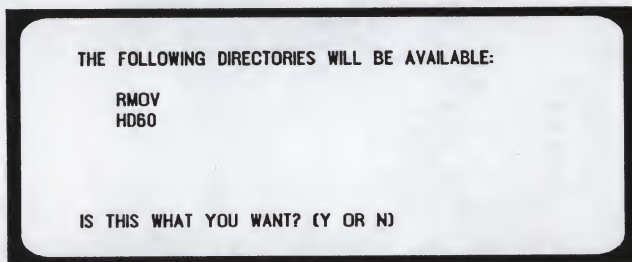
If you select **F** or **<Return>** in order to switch the floppy diskette and if your system has only 1 floppy disk drive ('F0'), the system will automatically select

that unit; otherwise, if your system has 2 floppy disk drives ('F0' and 'F1'), the system responds with:

ENTER UNIT ('CR' FOR 'F0')

For removable hard disks, on some systems the disk drive is designated H0, and on other systems, H1. You will enter the correct drive designation. In the sample screen, H0 is the removable hard disk. If you select the fixed hard disk drive by mistake, the system will assume that it is a removable disk. You will be allowed to enable, disable or switch directories on the fixed hard disk with this utility. However, DO NOT use this utility for the fixed hard disk because the *JPSD Utility is normally used for this purpose.

The system will display the directories configured for that unit which will become inaccessible when the disk is removed, and ask if this is what you want:



```
THE FOLLOWING DIRECTORIES WILL BE AVAILABLE:

RMOV
HD60

IS THIS WHAT YOU WANT? (Y OR N)
```

An <F4> response will return to the *UPSD screen. An N response will return to the previous prompt (ENTER UNIT NUMBER). A <Return> will default to a Y response.

Following a Y response, the system will disable the directories on that disk and respond with the following, depending upon which disk you are switching:

INSERT REMOVABLE HARD DISK AND 'CR' TO CONTINUE

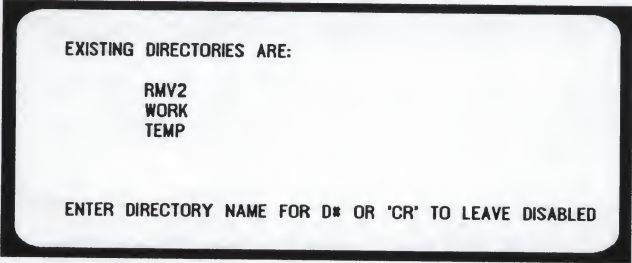
OR

INSERT DISKETTE AND 'CR' TO CONTINUE

A <Return> will indicate that you have loaded a disk. For removable hard disks, the system will respond with:

WAITING FOR DISK TO SPIN UP TO OPERATING SPEED

When the disk is ready, the system will check the directories on it and compare them with the logical disk numbers configured for that unit. In the example above, there are two logical disks (D0 and D7) configured for the removable hard disk (Unit H0), and one logical disk (D9) configured for the floppy drive (Unit F0). If the directories on the disk just loaded match the logical disks configured for that unit, the system will automatically log those directories onto the corresponding logical disks. If the loaded disk contains less or more directories than logical disks assigned to that unit, or the directory names do not match, the system will display the existing directories on the loaded disk and allow you to select which ones you want to log on, prompting you for each available logical disk:



EXISTING DIRECTORIES ARE:

RMV2
WORK
TEMP

ENTER DIRECTORY NAME FOR D# OR 'CR' TO LEAVE DISABLED

After you have selected directories for each logical disk configured for the unit, the system will change the directory accordingly, display the changes, and then return to the Utilities Menu.

Error Messages

ONLY EXISTING DIRECTORIES CAN BE SELECTED

At the ENTER DIRECTORY NAME prompt, an invalid name was entered.

NO DIRECTORIES ON THIS REMOVABLE HARD DISK, 'CR' TO CONTINUE

This indicates that the removable hard disk has not had a directory initialized. If so desired, use the *1PSD or *4PSD Utility to create a directory on this disk.

NO DIRECTORIES ON NEW FLOPPY, DISK IS NOW DISABLED

This indicates the floppy has not had a directory initialized.

DISK NOT UP TO SPEED; PLEASE CHECK FOR PROPER MOUNTING

This indicates that the removable hard disk probably has not been loaded properly. Reload and enter a <Return> to continue.

DISK RESERVED BY ANOTHER USER - CANNOT BE SWITCHED

This indicates the disk is still being used; it is reserved and cannot be switched.

D# CANNOT BE SWITCHED: FILES OPENED BY ANOTHER USER

This indicates the disk is still being used; files are open in a directory on this disk.

***RECOVER - Bad Track Sparing Utility**

Introduction

After a computer system has been installed and running for a period of time, it is not uncommon for sectors on the hard disk to become unreadable. There are many reasons for this to occur, including environmental contaminants (dust) and wear and tear on the physical disk drive. When a sector becomes unreadable or bad, any data it contained is lost. An ERROR 103 can be the result of such an unreadable sector. To prevent the errors associated with repeated attempts to write to these bad sectors, most operating systems allow for the read-dressing or sparing of bad sectors. For efficiency, many systems spare the entire track containing a bad sector. Tracks which contain one or more bad sectors are referred to as bad tracks.

At the time of installation, the *1PSD Utility executes bad track sparing on the hard disk. If a sector becomes unreadable after installation, the operator must use the *RECOVER Utility to spare bad tracks. Before running the utility, make sure you have two current backups of all important data. You may not be able to backup files containing bad sectors, but you may be able to copy any good sectors from the file to another file for later attempts at data rebuilding.

Features of this utility include: displaying or printing the Bad Track Table, testing for bad tracks, sparing bad tracks and printing a bad track impact report.

***RECOVER Requirements**

Prior to using the *RECOVER Utility, these requirements must be met:

- The utility must be run from the console.
- No one else may be on the system.
- The system printer (LP) must be available.
- TOS must have an existing Bad Track Table (BTT).
- The system must meet specific space requirements. More than 2 full tracks must be available as the last entry in the Available Sector Table (AVS). There must be 1 empty track (34 sectors) between the last entry in the AVS table and the beginning of the ATA. If these requirements are not met an error message is displayed. See "Space Requirements Notes" in the Error Message section for further information.

Abort Procedure

Before executing *RECOVER the operator should be aware that either the Escape Key or F4 will exit the procedure. If the Escape Key is used, the system interrupts the current process and allows you to abort or resume the procedure:

DO YOU WANT TO ABORT? (Y/N)

Enter Y to return to the Utilities Menu.

Execution of the *RECOVER Utility

At the system Utilities Menu:

ENTER SELECTION OR CR TO END: *RECOVER <Return>

If there are two hard disks, the system will prompt the operator to select the disk on which to run the utility. It then displays the "Display Bad Tracks" screen.

*RECOVER - BAD TRACK MAINTENANCE UTILITY 04/18/88

Disk: H0 Current Bad Tracks: 2 Available Bad Tracks: 77 Page 1
Display Bad Tracks ATA Start: 398752 Format Sector - Sector

184688-184721 313106-313139

<P>rint bad track table <A>dd bad tracks <C>hange display format
<T>est for bad <V>iew disk <D>own <U>p <Q>uit
tracks parameters Page page

Enter selection -->

NOTE: The default format is always sector-sector (starting sector/ending sector).

The first entry of the BTT is the first bad track spared in the ATA. Bad tracks are displayed on the screen from left to right and top to bottom. If the screen is filled and there is additional bad track data, enter <D> (Down Page) or <U> (Up Page) to scroll. A bad track in the ATA will be indicated by the message "BAD ATA."

To maintain the BTT, select one of the following bracketed letters:

- <P>rint bad track table
- <A>dd bad tracks
- <C>hange display format
- <T>est for bad tracks
- <V>iew disk parameters
- <D>own Page (scroll down)
- <U>p Page (scroll up)
- <Q>uit

<P> Print Bad Track Table

This function prints the BTT in the same format as shown on the screen. The following message appears:

PRINTING BAD TRACK TABLE TO LP....

<A> Add Bad Tracks

This function adds bad tracks to the BTT. The utility prompts:

ENTER SECTOR # OR <CR> TO QUIT:

The operator enters the information in the format chosen (see CHANGE DISPLAY FORMAT).

After the operator enters the information, the utility verifies that the entry is a valid sector. It then prompts:

PRINT FILE IMPACT REPORT? (Y/N)

This report indicates whether the bad sector is in a file directory, the system area or the AVS table. It can only be printed in the Sector-Sector format. The utility prompts:

SPARE BAD SECTOR? (Y/N)

At this point the operator can decide whether or not to spare tracks containing bad sectors. The utility then asks if another sector is to be added.

NOTE: Tracks that are on the DOS side of the partition or that contain the operating system cannot be spared.

<C> Change Display Format

This function changes the format of bad tracks from Sector-Sector to Head-Cylinder (or the reverse). This changes the method of adding bad track or bad sector information. The utility prompts:

SELECT DISPLAY FORMAT: <S>ECTOR NUMBER OR
<H>EAD/CYLINDER

<T> Test Disk for Bad Tracks

This function tests the hard disk for additional bad tracks. After disk testing is complete, the utility displays the "New Bad Tracks" screen and prints a new menu at the bottom of the screen. The screen contains the same information as the "Display Bad Track" screen except that it only displays those bad tracks found during the testing process. (If no new bad tracks are found, none will be shown.) The functions are:

- <P>rint new bad tracks
- <F>ile impact report
- <S>pare tracks
- <D>own page (scroll down)
- <U>p page (scroll up)
- <M>ain menu

<P> Print New Bad Tracks

This function prints only the newly found bad tracks. You may want to print this information before sparing the tracks.

<F> File Impact Report

This function creates a File Impact Report on all new bad sectors, and identifies which directories or files are affected. Output for this function is only to the system printer.

<S> Spare New Bad Tracks

This function spares the newly found bad tracks using the same procedure as the "<A> dd bad tracks" function on the main screen. The utility identifies the track sectors affected, the bad sector and the new sector containing the information. New bad tracks can only be spared once.

<M> Main Menu

This function exits the "New Bad Tracks" screen and returns to the main menu. If new bad tracks were found and spared, they will be displayed on "Display Bad Tracks" screen at the main menu. Bad tracks found but not spared will not be displayed at the main menu.

<V> View Disk Parameters

This function displays disk parameters similar to the following.

Disk Parameters for H0

Number of heads: 5

Number of cylinders: 999

Number of 256-byte sectors per track: 34

| | NAME | START | END | SIZE |
|----|------|-------|------|------|
| 1. | DOS | 0000 | 0770 | 0771 |
| 2. | T/OS | 0771 | 0998 | 0228 |
| 3. | | | | |
| 4. | | | | |

<Q> Quit the Display Bad Tracks Menu

ENTER SELECTION --> Q <Return>

Error Messages**ALL USER TASKS MUST BE OFF SYSTEM**

Only T0 can be on the system when running *RECOVER.

BAD TRACK TABLE NOT FOUND

A valid Bad Track Table (BTT) was not found. During installation the BTT is created when sparing is selected. If this is not done, it will not exist; TOS must be reinstalled and bad track sparing must be executed.

ERROR READING HARD DISK

Cannot read system information from the hard disk.

ERROR WRITING NEW BTT AND AVS INFORMATION TO HARD DISK!

An error occurred when writing to the hard disk. Corruption of the Bad Track Table and/or the AVS may result.

ERROR XX OCCURRED IN MODULE - XXXXXX

An unexpected error has occurred in the *RECOVER public program module specified.

ERROR XX OCCURRED ON LINE XXXX

An unexpected error occurred in the utility.

INCORRECT BAD TRACK TABLE FORMAT

The format or structure of the bad track table is corrupted.

INVALID CYLINDER #

The entered cylinder number is not in the TOS partition.

INVALID HEAD #

The entered head number is out of the range for this hard disk.

MAXIMUM NUMBER OF BAD TRACKS EXCEEDED!

More bad tracks have been found than allowed by this version of TOS. The maximum size of the BTT is fixed according to the version of TOS that has been installed.

NEW BAD TRACKS HAVE ALREADY BEEN SPARED!

Bad tracks can only be spared once. An attempt to add a bad track already spared results in this message.

NO HARD DISKS CONFIGURED

The system has no hard disks configured.

NOT ENOUGH AVAILABLE SECTORS TO SPARE TRACK

The AVS is too small to spare any tracks. See "Space Requirements Note."

ON TRACK WHICH HAS BEEN SPARED! (HD = XX, CYL = XXX)

The entered bad track or bad sector has already been spared.

PRINTER NOT READY

The system printer (LP) is not available or on-line.

SECTORS IN FRONT OF ATA NOT AVAILABLE FOR SPARING!

In order for *RECOVER to spare a track, there must be 1 empty track (34 sectors) between the last entry in the AVS table and the beginning of the ATA. See "Space Requirements Note."

SECTOR: XXXX IN SYSTEM AREA NOT SPARED!

The indicated sector is in the operating system area and cannot be spared.

WRONG BAD TRACK TABLE LENGTH IN TOC

The length of the bad track table does not match the version of the operating system.

Space Requirements Note

Use the following information to verify and correct a space requirements problem.

More than 2 full tracks must be available as the last entry in the AVS

- From the utilities menu run *RECOVER.

- Select <V> (View disk parameters).
- Note the number of sectors per track (e.g. 34).
- Return to the utilities menu and run *LPSD.
- Select a listing by Available Sectors. If this number (e.g. 1357) is more than twice the number of sectors per track, this requirement has been met.

There must be 1 empty track (34 sectors) between the last entry in the AVS table and the beginning of the ATA.

- From the Utilities Menu run *LPSD, and select a listing by Available Sectors.
- Find the last entry, add the length to the starting sector and note for future reference (e.g. $38345 + 313 = 38658$).
- Run *RECOVER and subtract the above total from the ATA starting sector listed on this screen (e.g. $38692 - 38658 = 34$).
- The remainder obtained in the previous step must equal the number of sectors per track for your system (e.g. 34 sectors per track = 34 sectors in front of the ATA).

If you do not have the required space as calculated above you must do the following:

- Use *LPSD to list file locations by Location.
- Determine what file is located in the 1 track (34 sectors) in front of the ATA and move or remove the file.
- If no file is found in this area, the system may have a gap in the AVS table. DISPAK should be used to correct this situation.

